In the Claims:

Please cancel Claims 2-3, 5, 9, 12-13, 15, 19, 22-23, 25 and 29; amend Claims 1, 4, 11, 14, 21 and 24; and add new Claims 31-33, all as shown below. Applicant respectfully reserves the right to prosecute any originally presented or canceled claims in a continuing or future application.

1. (Currently Amended): A system for organization of software application files during development [[a]] and subsequent deployment of the software application to a server ~~software development process~~, comprising:

a split directory structure stored on a computer medium that stores files for a software application, wherein the split directory structure includes both a source folder that stores editable source files ~~for use with or~~ as part of [[a]] the software application, [[;]] and a corresponding output folder that stores compiled files as part of the software application, and wherein the split directory is accessed as a virtual JAR file that provides an abstraction over the two folders therein;

a server upon which the software application will be deployed; and

a deployment tool that allows the user to specify the output folder during deployment of the software application, wherein during the deployment the server recognizes the split directory structure and deploys the application by making requests to the virtual JAR file which checks both the source folder and the corresponding output folder for software application files, before deploying the software application files to the server

~~an output folder that stores compiled files for use with or as part of said software application; and,~~

~~wherein said source folder and said output folder output folder form a split directory for use in deploying said software application~~.

2-3. (Canceled).

4. (Currently Amended): The system of claim 1 wherein the output folder includes a file that identifies the output folder as being part of [[a]] the split directory which also includes the corresponding source folder.

5. (Canceled).

6. (Original): The system of claim 1 wherein said software application, or another software application can point to the output folder to access or retrieve resources in either the output folder and/or the source folder as necessary for operation of the software application.

7. (Original): The system of claim 1 wherein said output folder is automatically created and populated upon compiling the software application.

8. (Original): The system of claim 1 wherein said output folder can be deleted to remove the latest build of the software application, and then recreated to create a new build.

9. (Canceled).

10. (Original): The system of claim 1 wherein the source folder is populated with source files that are stored in or retrieved from a source control system.

11. (Currently Amended): A method for deploying a software application to a server ~~organizing and using source and output files during a software development process~~, comprising the steps of:

storing files for a software application in a split directory structure on a computer medium, wherein the split directory structure includes both a source folder that stores editable source files as part of the software application, and a corresponding output folder that stores compiled files as part of the software application, and wherein the split directory is accessed as a virtual JAR file that provides an abstraction over the two folders therein; and

allowing the user to specify the output folder during deployment of the software application to the server;

wherein during the deployment the server.

recognizes the split directory structure by making requests to the virtual JAR file which checks both the source folder and the corresponding output folder for software application files, and

deploys the software application files to the server

~~providing a source folder that stores source files for use with or as part of a software application;~~

~~providing an output folder that stores compiled files for use with or as part of said software application;~~

~~recognizing said output folder and the contents stored therein as being part of a split directory for use in deploying said software application; and,~~

~~identifying both said source folder and said output folder as a split directory for use in deploying the application~~.

12-13. (Canceled).

14. (Currently Amended): The method of claim 11 wherein the output folder includes a file that identifies the output folder as being part of [[a]] the split directory which also includes the corresponding source folder.

15. (Canceled).

16. (Original): The method of claim 11 wherein said software application, or another software application can point to the output folder to access or retrieve resources in either the output folder and/or the source folder as necessary for operation of the software application.

17. (Original): The method of claim 11 wherein said output folder is automatically created and populated upon compiling the software application.

18. (Original): The method of claim 11 wherein said output folder can be deleted to remove the latest build of the software application, and then recreated to create a new build.

19. (Canceled).

20. (Original): The method of claim 11 wherein the source folder is populated with source files that are stored in or retrieved from a source control system.

21. (Currently Amended): A computer readable medium including instructions stored thereon which when executed cause the computer to perform the steps of:

storing files for a software application in a split directory structure on a computer medium, wherein the split directory structure includes both a source folder that stores editable source files as part of the software application, and a corresponding output folder that stores compiled files as part of the software application, and wherein the split directory is accessed as a virtual JAR file that provides an abstraction over the two folders therein;

allowing the user to specify the output folder during deployment of the software application to the server;

recognizing the split directory structure by making requests to the virtual JAR file which checks both the source folder and the corresponding output folder for software application files; and

deploying the software application files to the server

~~providing a source folder that stores source files for use with or as part of a software application;~~

~~providing an output folder that stores compiled files for use with or as part of said software application;~~

~~recognizing said output folder and the contents stored therein as being part of a split directory for use in deploying said software application; and,~~

~~identifying both said source folder and said output folder as a split directory for use in deploying the application~~.

22-23. (Canceled).

24. (Currently Amended): The computer readable medium of claim 21 wherein the output folder includes a file that identifies the output folder as being part of [[a]] the split directory which also includes the corresponding source folder.

25. (Canceled).

26. (Original): The computer readable medium of claim 21 wherein said software application, or another software application can point to the output folder to access or retrieve resources in either the output folder and/or the source folder as necessary for operation of the software application.

27. (Original): The computer readable medium of claim 21 wherein said output folder is automatically created and populated upon compiling the software application.

28. (Original): The computer readable medium of claim 21 wherein said output folder can be deleted to remove the latest build of the software application, and then recreated to create a new build.

29. (Canceled).

30. (Original): The computer readable medium of claim 21 wherein the source folder is populated with source files that are stored in or retrieved from a source control system.

31. (New): The system of claim 1 wherein the virtual JAR file first checks the source folder for the software application files including any classes or resources needed by the software application, and, if the classes or resources are not found in the source folder, then checks the output directory.

32. (New): The method of claim 11 wherein the virtual JAR file first checks the source folder for the software application files including any classes or resources needed by the software

application, and, if the classes or resources are not found in the source folder, then checks the output directory.

33. (New): The computer readable medium of claim 21 wherein the virtual JAR file first checks the source folder for the software application files including any classes or resources needed by the software application, and, if the classes or resources are not found in the source folder, then checks the output directory.